Introduction to the Internet of Things Final Project

Seeing Personal Environmental Health Risk

Yu-Chun Lin
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan
b10901151@ntu.edu.tw

Hao-Kai Chi
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan
b11901133@ntu.edu.tw

***Abstract*—Environmental conditions have significant affect on one's health. To better evaluate the health risk from environmental conditions individually, and Internet of Things based system is proposed. The system involves end points for collecting air quality and heart rate, gateway, backend and a mobile app displaying the output results.**

***Index Terms*—Internet of Things (IoT), Personal Health, Environmental Health Risk, Smart Watch, Bluetooth, MQTT, Arduino, Raspberry Pi, MIT APP Inventor, PM2.5, Resting Heart Rate.**

I. Introduction

We know that different weather conditions may lead to different consequences for different individuals. However, such differences within human beings still remain unconscious in the aspect of wearable smart devices, such as smartwatches. The products on the market are in the absence of quantifying individuals' reactions toward the potential health risks from the surroundings. Even with a similar environmental condition, people may suffer from extremely distinctive consequences. Without taking personal differences into account, those products still lack personalization and customization. In this demonstration, we focused on the impact of air quality on heart rate. Although there are many environmental factors affecting heart rate, including but not limited to temperature and humidity, air quality is an important yet often neglected factor. Almost all people know that poor air quality has a negative impact on health, however, most people only notice the negative impact on the respiratory system, unaware of the influence of air quality on heart rate.

To address the aforementioned issue, we manage to gracefully utilize user history data through the hierarchical system structure, which not only secures customers' information but also efficiently builds up our basis for personalized normalization. Combined with an application interface, we further improve the controllability and visualization of information based on regression computation. Enhanced approaches can be further promoted by integrating large amounts of data collected with standardized metrics as a normalization baseline for more accurate analysis and prediction.

II. System Structure

In this section, we will introduce our system four-folded. From the acquisition of heart rate and air quality data to the overall structure: gateway, backend, and application interface.

A. Heart rate and air quality acquisition

In our system, obtaining online and offline data are both important. With sufficient offline data, we can conduct regression analysis with higher precision. On the other hand, online data is crucial for analyzing the potential risks of the current surroundings, based on users' history data, for real-time usage in the broad sense. However, it's also important to note that the idea of "real-time" here is not about the range of seconds or minutes. The "real-time" here is about collecting data from a period of time, say an hour for instance, and calculating the mean within the specific unit of time of consideration. The calculated values can then be useful for visualization and analysis to further showcase the users' reactions and the effects of the environments on the person.

With this concept in mind, it's obvious that the true essence of our work is to capture the implicit threat of potential environmental risks that are easily ignored by human beings. We are not aiming at immediately discovering variations of heart rates that do not follow the general distribution and fluctuate abnormally. Instead, we hope to provide an inspection over a slightly longer period of time to fetch the insignificant yet lethal hazards that are still uncovered, especially in the design of smart devices nowadays. With our system, we hope to illustrate the potential environmental risks to humans that should be aware of to attain a healthier life.

To build the personal health risk model, the heart rate between 2 to 6:00 a.m. and the air quality between 0 to 4:00 a.m. are extracted. The data is then used to evaluate the risk. there are so many factors affecting a person's heart rate and it may be difficult to extract the relationship between heart rate and air quality due to fluctuations in heart rate. However, during night time, since the person is sleeping, there is no way that physical activity affects heart rate. However, fluctuations in heart rates still hinder the extraction of the relationship between heart rate and air quality. To solve the problem we use the hourly mean heart rate. Most of the fluctuations are averaged and it is easier to observe the relationship.

Considering the data quality, we turned to Climate Observation Data Inquire Service by Central Weather Administration [1], and Taipei Environmental Investigation and Analysis Cen-

ter [2] for the backend to build up fundamental structure to support our extended parts of the system. The data provided by [1], and [2] are based on standardized regulation, where the results would be feasible and trustworthy in our case that safety is our primary consideration as well. Any misleading information might result in misunderstanding; thus, we carefully chose these data to implement our analysis. With their publicity and accuracy, we can then emphasize how to properly utilize them to our task.

The heart rate is collected by using smartwatch Garmin Vivosmart 5. This device can be used to evaluate a person's heart rate. One of the merits of using a smartwatch is its easy access to the device. One of the drawbacks of the Smartwatch is there is no wireless connection of the watch. Although the Smartwatch provides a Bluetooth connection function it does not allow raw data output. The best method to get the raw data is by emailing Garmin requesting raw data. Needless to say, this is very inefficient. Alternatively, a USB transmission line can be used to transmit the data. The method may be a viable solution, but the data reading time is very long if there are a lot of data. Meanwhile, the data are messy and difficult to analyze. Both methods have significant drawbacks.

B. Gateway

Our gateway is a combination of Arduino Uno and Raspberry Pi 4. We utilize both of their merits to attain our goals. We use Arduino Uno for the Bluetooth connection and MQTT connection, where we receive user instructions through Bluetooth from the APP interface and send information to the application with the aid of MQTT. As for Raspberry Pi 4, its main role is to be the gateway for the interconnection with our backend. The interconnection of our gateway of Arduino and Raspberry Pi is realized by a USB cable with the ACM port of Raspberry Pi and the USB port of Arduino. The details of each part of the above general description will be listed and discussed below:

- From APP to Arduino:
  We utilize MIT APP inventor as the foundation of our application interface. By clicking the buttons and entering information such as your location (longitude), password, and user ID, the data will be transmitted through Bluetooth. We select HC-05 Bluetooth module as our approach to set up the connection between Arduino and our cellphone.
  As depicted in Figure 1, we connect our HC-05 module to Arduino and set the serial transmission rate to 9600. After processing the input signals from APP and Bluetooth, we analyze the properties of the signals and then conduct the corresponding actions.
- From Arduino to Raspberry Pi:
  In our system, we use GP2Y10 PM2.5 sensor as our simulation for the in-time data fetching mechanism. The wiring of our design follows [4] is demonstrated in Figure 2 below. The dust sensor connects to the A0 pin, and the signals will go through an analog-to-digital converter to map the values to voltages ranging from 0 V to 5 V, as shown in (1). We then follow the linear equation and unit conversion to measure the concentration of the PM2.5 rounding to 2 decimal places in $\mu g\,m^{-3}$.

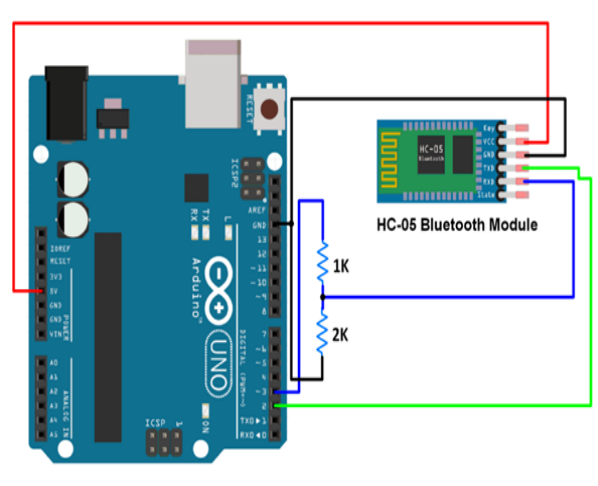

Fig. 1. Illustration of our wiring for HC-05 on Arduino Uno from [3].

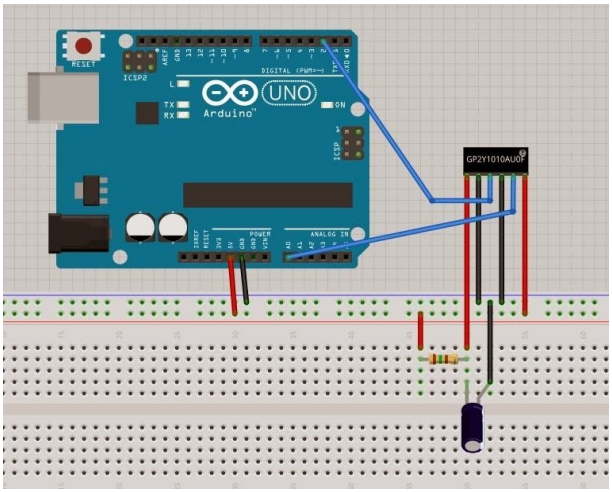

Fig. 2. Illustration of our wiring for GP2Y10 PM2.5 sensor on Arduino Uno from [4].

We have slightly modified a part in Figure-2 of our prototype, where we replace the 150Ω resistor with four parallel 1kΩ resistors equivalent to 250Ω. Thus, we rewrote our linear equations to correctly calculate PM2.5 based on [5], as in (2), with some changes to output our final results to the Raspberry Pi. The connection between Arduino Uno and Raspberry Pi is simply using a USB cable with the USB and ACM ports of Arduino and Raspberry Pi, respectively.

$$calcVoltage = voMeasured \times \frac{5.0}{1024.0} \quad (1)$$

$$dustDensity = 0.17 \times calcVoltage - 0.1 \quad (2)$$

- Send data to backend:

Despite data is continuously collected in GP2Y10 module, the messages received from the APP to Arduino via Bluetooth will decide which mode the whole gateway system should be in. For instance, mode 1 is when the user is called for backend data without the need to forward the PM2.5 results to Raspberry Pi for further usage; while mode 2 is when the user specifies that the PM2.5 data will be given from the Arduino and Raspberry Pi should utilize the data to conduct the rest of the computation and processing. There are also other modes where the gateway will deal with data in different ways and interact with our backend in different manners. For the detailed functionality of these instructions from users, please refer to subsection D, APP, in this section.

The overall connection from the gateway to the backend and the reverse are based on MQTT. To send data to the backend, the Raspberry Pi will subscribe to the topic "hello/group1/rpi2pc" and publish the data to the backend. In mode 1, the Raspberry Pi will send a request to the backend for the latest data stored in the backend database. The csv file with the corresponding station to fetch is determined by the longitude that the user originally typed into the application interface. The data of such positional information will, thus, be sent to the backend to search for any matching file and the newest information.

As for mode 2, the logic block that controls whether to send real-time PM2.5 data to the Raspberry Pi will be open. The Raspberry Pi will not only use these data to compute linear regression for risk analysis locally in the gateway, but also publish the PM2.5 information, position or station information, and time to the backend for the database to properly deal with these data. The useful data will then be stored in corresponding csv files for further and larger usage. We can then figure out an even more up-to-date norm with the help of such a renewable database.

- Receive data from backend:
  After sending data from the gateway to the backend, it's crucial to receive data from the backend to the gateway as well. With previous discussions, there are mainly two modes that account for the consideration in this section. In mode 1, the Raspberry Pi will request data from the backend. After receiving the request, the backend will process the request and send back the required data. Thus, by subscribing to the topic "hello/group1/pc2rpi", Raspberry Pi can obtain the data that are indispensable for calculation.

  As for mode 2, we do not explicitly use the "acknowledgment" for higher quality of service here. As mentioned above, we use MQTT for publishing data and do no additional checks with multiple handshaking. We print out information on Raspberry Pi screen and the backend screen (i.e. computer screen) if data are sent or received at either side of the communication.
- From Raspberry Pi to Arduino:
  After the interactions with the backend during either mode 1 or mode 2, the system will be in mode 3, which is also determined by the user end. In mode 3, the logical block controlling the transmission of PM2.5 information will be closed so that the PM2.5 values will stop sending to the Raspberry Pi.

  Then, the Raspberry Pi will start calculating the potential environmental risks. The analysis pipeline consists of several key processing stages designed to handle temporal data alignment and statistical analysis. Initially, the heart rate data undergoes preprocessing that converts timestamp information into discrete temporal components (month, day, hour, minute) and calculates hourly averages of heart rate measurements. Concurrently, other data also undergoes similar temporal decomposition but maintains its original granularity without averaging.

  The core of the analysis focuses on the early morning period (2-6 AM), where this time window reveals more pronounced relationships between air quality and physiological responses during sleep. This is because only resting heart rate can show the true effect the environmental conditions may have on human beings according to X. Xie et al. [6]. A crucial design decision in our implementation is the temporal alignment mechanism, which matches heart rate measurements with the previous hour's air quality data, allowing for the investigation of potential delayed effects of air quality on heart rate patterns.

  We employ a matrix-based approach for efficient handling of the temporal data series. The matrix_processor function creates paired observations by matching heart rate data with corresponding air quality measurements based on temporal proximity.

  In the statistical analysis component, we utilize both linear regression and correlation analysis to quantify the relationship between air quality and heart rate. The implementation leverages the Linear_LS_Regression solver for computing the regression coefficients, while also calculating the Pearson correlation coefficient and its associated p-value to assess statistical significance. This approach can provide multiple metrics for evaluating the strength and reliability of the observed relationships.
- From Arduino to APP:
  After the calculation method described above, we will use parameters that we get to further distinguish the risks to the user in a broad sense. We classify the risks into three classes, namely nonsensitive, low risk, and high risk. We make such classification based on the threshold of a single parameter or the combination of multiple parameters altogether.

  In mode 4, we will publish the topic "result" with the risk results(formatted in 0, 1, or 2), and the PM2.5 values. The APP will subscribe to the topic and, thus, receive the processed information from the gateway back to the user end. The APP we wrote will finally visualize the results that users can perceive gracefully at ease.

C. Backend

We simulate the backend architecture using a personal computer. The backend will subscribe and public data respectively in topics ”hello/group1/rpi2pc”, and ”hello/group1/pc2rpi”. It will also manage a directory containing several csv files that are stored with history data of lots of different positions from anywhere, and at any time.

The backend provides the service that it can send specific PM2.5 information of the nearest position to users to the gateway after receiving the request. It can also update its database from user data during mode 2. With the multi-mode functionality, our backend can enhance over time by obtaining and processing more and more user data. The increasing accuracy of the risk prediction will further improve the possible health issues discovery. This is as well our ultimate goal to show people the hidden risks of environmental conditions to the health and encourage people to seek medical treatment as soon as possible if the conditions are harmful.

D. APP

The mobile application enables the user to access the air quality data and personal risk. The user interface of the app is shown below. First, the user must log in by entering the correct username and password and ticking “I am not a robot”. This process enables the app to verify the user and fetch the personal risk data. Each account is associated with a set of risk parameters. By logging in, the personal risk parameter is used to calculate the risk. Meanwhile, this process also ensures data security by preventing unauthorized access. The “I am not a robot” questions prevent logging in using a computer program.

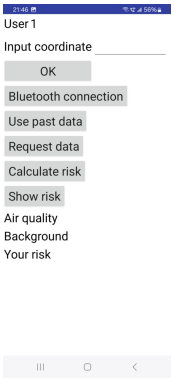

Fig. 3. The log in user interface of the app.

After successfully logging in, the app then asks permission to access the location service if the user has not consented previously. To function properly, the access must be granted. Ideally, the app will decide the location of the user and find the nearest air quality station. However, in our demonstration example, the user always stays at the same place so the location function is activated by typing the coordinate (degree longitude) instead. Bluetooth function must be activated to transfer the data to the gateway (Arduino). The user must pair the Bluetooth device before using the app. To ensure

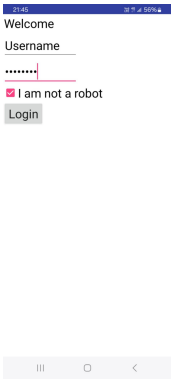

Fig. 4. The main user interface of the app.

a successful Bluetooth connection, the user needs to choose the Bluetooth address from a list of paired devices (usually there is only one). Then, the user can choose to use past data (Data stored in the backend) or request current data (Data sent from the nearest air quality station) by clicking the respective buttons. Usually, the request current data function is used so that the real time data can be used to calculate the risk. However, in some certain cases, real time data is not available, for example, due to a malfunctioning quality sensor. In this case, the backend data, which includes the latest input from the air quality sensor, can be used. After choosing the type of data used for calculating the risk, the calculate risk button is pressed. A message is sent via Bluetooth to Arduino and Raspberry Pi to ask Raspberry Pi to calculate the risk. Meanwhile, the MQTT function on the mobile device is also activated. The MQTT function will turn off after one minute. Finally, the display risk button is pressed, and a prompt is sent to Raspberry Pi. Raspberry Pi returns the air quality and the risk data to the mobile device via MQTT [7].

After receiving the air quality and risk data, the app shows the result then the risk is categorized into three categories. The first category is nonsensitive in which the user’s heart rate is not sensitive to the air quality. The second category is low risk. This indicates that the user’s heart rate is sensitive to the air quality, however, the air quality is good, so there is no significant impact on the user’s heart rate. The third category is high risk. This means there is a significant risk to the user’s heart rate. The user can make use of the messages when making decisions for daily activities, for example, the user may refrain from jogging if the air quality is poor and there is a significant health risk for the user. If the user is in a room, the user may choose to turn on the air cleaner.

The app is developed using MIT App Inventor. MIT App Inventor provides a platform for programmers to develop an app with built in user interface and graphical programming functions. Although it is easy to build an app using MIT App Inventor, the performance for some of the source codes does not perform well. For example, the Bluetooth function has a significant delay when sending and receiving messages. It takes a long time for Arduino to receive the message sent

from a mobile device. Although this delay does not impact significantly when sending data, for receiving, the situation is different. Arduino sends the message only once. If there is a significant delay in receiving a message, the received message function may not be activated when the message is sent. This causes significant errors. To alleviate this issue, MQTT is used when sending the data from Raspberry Pi to the mobile device. Since the mobile device subscribes to MQTT for 60 seconds, if there is any message within 60 seconds, the message can be received by the mobile device correctly.

III. Comparison and Future Outlook

Our approach has leveraged lots of methods to the consideration of the scalability and integration of other modules to our structured system. In this section, we will discuss our merits and future outlook. We hope our design can be generally described as a prototype for future products.

A. Comparison

As depicted in aforementioned paragraphs, our design values the scalability and data utilization rate.

TABLE I
Comparison of Current Status and Proposed Solution

	Data Utilization Rate	**Readability**	**Extensibility**	**Practicality**
Current	Lower	Numerical values without comparison to population data	Limited utility for medical-related data construction	Too simplified and lacks follow-up data processing support
Our's	Higher	Considers both regular patterns and individual influence factors in analysis	Easily extensible modular structure	Data has both unified and distributed factor analysis, with layered comparison implementation

From Table 1, we highlight that we not only use history data to further increase our database, but also let these data be used for additional analysis on risk prediction. The more the data, the further enhanced accuracy we can make. We are continuously pursuing a convergence to the statistics that have considered every human being in the globe. Though it's not attainable, our method can make us closer to the promising distribution which is acceptable for clinical usage.

Besides, our method has created a concise yet informative interface, where users can be aware of their circumstances through it. And the information provided by our system can guide people to discover the potential risk that they might not have discovered. The risk that results from environmental conditions is still absently undiscovered. Without our approach, the connection between self-health and the environmental condition is as well inconceivable. Thus, our system has its value in providing a precaution for potential risks that are, in a sense, invisible.

Last but not least, the design of our system can be scaled up easily with other modules. For instance, since we are limited to using Garmin Vivosmart 5 in our final project, the data transmission rate and the wired requirement have set a boundary for our attempt. However, our system can be updated by simply replacing the wired transmission of wireless protocol to send data from smartwatches to the gateway or even other parts of our structure. This can greatly enhance the convenience of using our products. Also, it's also feasible to use more database systems such as SQL or No-SQL to further improve the data storing mechanism in the backend. By utilizing these large scale, existing methods, our prototype can be scaled up to become a lot more robust, accurate, and data efficient.

B. Future Outlook

This system we propose for demonstration purposes only and the real system must take several other factors into account. The most significant factor we neglected is the acquisition of heart rate by a wireless system. Due to the limitation of the smart watch, we could not implement this part and we needed to use a computer to process the data previous to the application. With a wireless connection function, all the data processing can be done in the gateway via Raspberry Pi. Meanwhile, we only have two air quality sensors and the amount of the data that we have is rather small, before aiming for commercial application the amount of data must be large enough. Currently, there are many air boxes sensing air quality and it can be used to increase the amount of air quality data. However, we must notice that an air box is less precise than the air quality observation network and the setup of an air box may not follow the standards. Finally, besides air quality, there are still several other very important environmental factors that will affect heart rate. We can add those parameters into account.

IV. Value Evaluation

The class file is designed for, but not limited to, six authors. The system is most useful for people with heart-related chronic diseases since a high heart rate poses a significant impact for people with heart-related chronic diseases. However, the system is also beneficial for healthy people. They can know their risk before the negative impacts of air pollution occur. Heart rate can be easily acquired by a smartwatch, which is not expansive and provides more function than sensing heart rate. It is also possible to integrate some of the functions into an air box. If it is possible to incorporate both functions together, then large scale application can be achieved. For example, a company selling smart watches may choose to cooperate with air quality observation network and remind the user of their

risk. Such an application is not expensive to build because there are already a lot of smartwatches and air quality data available. Some of the air quality data are open source and can be used free of charge. Therefore, the cost is similar to a smartwatch but this feature can increase the value of a smartwatch. There about 45.5 millions smartwatch users all over he globe up to 2024 [8]. In addition, we know that people are willing to and are interesting in buying smartwatches that are equipped with health tracking function [9]. The promising statistics (68% and 80% for Spanish and Turkish citizens) highlights the flourish future of our idea. Although it is difficult to predict the exact cost and return of the system, it is plausible to assume that the product with little additional cost and some potential value is profitable.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to Dr. Chi-Hsien Chen for providing valuable technical discussions and insights throughout our project. We are deeply thankful to Professor Alexander (Alex) I-Chi Lai and Professor Ruey-Beei Wu for their guidance and instruction throughout the courses. We would also like to thank the teaching assistants for their technical support and assistance with components.

REFERENCES

[1] CODIS Climate Observation Data Inquire Service. [Online]. Available: https://codis.cwa.gov.tw/StationData

[2] Taipei Environmental Investigation and Analysis Center. [Online]. Available: https://www.tldep.gov.taipei/Public/DownLoad/AqiHour.aspx

[3] Bluetooth Module HC-05 Pinout, AT Commands & Arduino Programming. [Online]. Available: https://www.electronicwings.com/sensors-modules/bluetooth-module-hc-05-

[4] GP2Y10 PM2.5 dust sensor. [Online]. Available: https://sites.google.com/site/wenyumaker/

[5] Application Guide for Sharp GP2Y1026AU0F Dust Sensor (Arduino Uno/Mega).

[6] X. Xie, Y. Wang, Y. Yang, J. Xu, Y. Zhang, W. Tang, T. Guo, Q. Wang, H. Shen, Y. Zhang, D. Yan, Z. Peng, Y. Chen, Y. He, X. Ma. Long-term exposure to fine particulate matter and tachycardia and heart rate: Results from 10 million reproductive-age adults in China

[7] How to make MQTT android application using MIT app inventor. [Online]. Available : https://highvoltages.co/iot-internet-of-things/mqtt/how-to-make-mqtt-android-application-using-mit-app-inventor/

[8] How Many SmartWatch Users Are There? [Online]. Available: https://www.statista.com

[9] Wearable Technology - Market Share Analysis, Industry Trends & Statistics, Growth Forecasts (2024 - 2029). [Online]. Available: https://www.gii.tw/report/moi1549712-wearable-technology-market-share-analysis-industry.html

[10] Future of smart devices. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2023/10/19/2024-iot-and-smart-device-trends-what-you-need-to-know-for-the-future/

[11] Garmin data application platform. [Online]. Available: https://www.garmin.com/zh-TW/account/datamanagement/

We have used ChatGPT to assist developing and debugging our source codes.